Caroiaj – Algoritmul lui Lee

#include <fstream>

#define DimMax 20 #define DimMaxCoada 400 using namespace std;

int dx[4]={-1, 0, 1, 0}, dy[4]={0, 1, 0, -1};

struct Element

{ int l,c; //pozitia in caroiaj

unsigned d; //distanta minima pana la pozitia l,c } ;

Element C[DimMaxCoada], x, y;

int A[DimMax][DimMax], n, m, x0, y0, i, j, k, IncC, SfC;

int main()

{ ifstream fin("careu.in");

fin >> n >> m;

fin >> x0 >> y0;

for (i=1;i<=n;i++)

for (j=1;j<=m;j++) A[i][j] = -2;

while (fin)

{ fin >> i >> j; //citesc coordonatele obstacolelor

A[i][j] = -1; } //marchez obstacolele cu -1

fin.close();

//BORDARE

for (i=1;i<=n;i++) A[i][0]=A[i][m+1]=-1;

for (i=1;i<=m;i++) A[0][i]=A[n+1][i]=-1;

//INITIALIZARE coada

x.l=x0; x.c=y0; x.d=0; A[x0][y0]=0; C[IncC]=x;

//algoritmul lui LEE

while (IncC <= SfC) //parcurg COADA

{

x = C[IncC++]; //EXTRAG un element din coada

//ne deplasam in cele 4 directii

for (k=0; k<4; k++)

{ y.l = x.l + dx[k]; y.c = x.c + dy[k];

if (A[y.l][y.c] == -2) //y- pozitie libera

{

y.d = x.d + 1; A[y.l][y.c] = y.d;

C[++SfC]= y; //duc y in coada

}

}

}

//afisare solutie

ofstream fout("careu.out");

for (i=1;i<=n;i++)

{ for (j=1;j<=m;j++) fout<<A[i][j]<<" ";

fout<<'\n';

}

fout.close();

return 0;

}